

PERANCANGAN DAN PEMBUATAN WORD GAME SCRAMBLE DENGAN ALGORITMA GENETIKA

Anita Qoiriah

Teknik Elektro, Fakultas Teknik, Universitas Negeri Surabaya, anitaqoi@yahoo.com

Veri Andriyawan

D3 Manajemen Informatika, Fakultas Teknik, Universitas Negeri Surabaya

Abstrak

Banyak metode pembelajaran yang digunakan guru agar murid dapat menangkap apa yang sedang diajarkan, salah satunya adalah dengan permainan. Metode pembelajaran ini mampu menyampaikan tujuan pokok belajar dengan cara yang menyenangkan, sehingga para siswa diharapkan dapat lebih bersemangat dan tertarik terhadap topik pembelajaran. Metode pembelajaran sambil bermain juga dapat diterapkan pada salah satu pelajaran penting di sekolah, yakni pelajaran Bahasa Inggris. Permainan berbahasa Inggris ini dikenal dengan Word Games. Dari sekian banyak macam jenis permainan Word Games yang ada, Scramble termasuk salah satu jenis permainan Word Games yang populer. Dalam pembuatan Word Game Scramble diterapkan pengambilan data dari hasil metode yang dilakukan oleh Andrea Krause berdasarkan Hidden Markov Models (HMM) dan Algoritma Genetika untuk proses pengacakan yang optimum dalam susunan huruf yang diperoleh, dengan memanfaatkan nilai fitness untuk mencari individu terbaik. Dari nilai fitness tersebut akan dilakukan lagi proses seleksi, proses cross over dan proses mutasi sehingga individu yang diperoleh benar-benar individu terbaik yang akan membentuk suatu susunan huruf-huruf alfabet dalam board 4x4. Hasil yang diperoleh dari proses ini adalah individu terbaik yang akan membentuk suatu susunan huruf-huruf alfabet dalam board 4x4 dan untuk membuat susunan huruf dalam board 4x4 yang optimum diperlukan pembobotan huruf sesuai dengan hasil penelitian yang dilakukan oleh Andrea Krause berdasarkan Hidden Markov Models (HMM) dalam pengisian nilai fitness pada proses algoritma genetika. Sehingga susunan huruf dalam board 4x4 menjadi lebih optimum dibandingkan dengan penelitian sebelumnya, agar dapat mempermudah pemain (user) untuk mencari kosakata dalam board 4x4.

Kata kunci: *Word Games, Scramble Games, Algoritma Genetika, Hidden Markov Models(HMM)*



PENDAHULUAN

Dalam pelajaran bahasa Inggris terdapat metode pembelajaran berupa permainan yang bisa mengasah, menganalisis, menghafal sebuah kosakata bahasa Inggris sekaligus bisa juga sebagai ajang *refreshing* siswa dalam belajar dengan cara bermain *Word Games*. Metode pembelajaran ini mampu menyampaikan tujuan pokok belajar dengan cara yang menyenangkan, sehingga para siswa diharapkan dapat lebih bersemangat dan tertarik terhadap topik pembelajaran.

Scramble termasuk salah satu jenis permainan *Word Games* yang populer yang terbentuk dari susunan *board*. Terdapat berbagai permasalahan dalam pembuatan *scramble* ini seperti pengacakan huruf-huruf untuk *generate board*, pencarian kata pada board 4x4 oleh komputer, dan pencocokannya kata pada kamus beserta sistem penilaiannya.

Penelitian yang pernah dilakukan oleh Noswa Sabdifha (2010) tentang perancangan dan pembuatan aplikasi word game scramble pada board dua dimensi didapatkan hasil algoritma depth first search sudah dapat melakukan pencarian mendalam pada board 4x4 untuk menemukan seluruh kata yang terdapat didalamnya. Namun untuk fungsi randomization atau pengacakan kata pada board 4x4 tidak dapat menghasilkan kosakata sebanyak yang diharapkan, sehingga menyulitkan pengguna untuk menemukan lebih banyak kosakata.

Tahap pengacakan *board* 4x4 dilakukan pengacakan secara biasa sehingga kosakata pada *board* 4x4 yang dihasilkan tidak optimum dan bisa terjadi pada tahap pengacakan tersebut tidak terdapat kosakata yang dihasilkan sama sekali.

Berdasar penelitian tersebut, maka dikembangkan pada penelitian ini yaitu bagaimana merancang dan membuat *word game scramble* dengan algoritma genetika pada tahap pengacakan huruf.

Dalam *game scramble* ini akan digunakan sebuah algoritma genetika, dimana dalam penerapan algoritma genetika untuk mendapatkan suatu pencarian kata yang lebih banyak dalam pola pengacakan kata dalam suatu *board* 4x4. Dan data yang digunakan mengambil dari hasil metode penelitian yang dilakukan oleh Andrea Krause berdasarkan *Hidden Markov Models* (HMM), agar susunan kata dalam board 4x4 menjadi optimum dan lebih banyak kosakata yang dihasilkan.

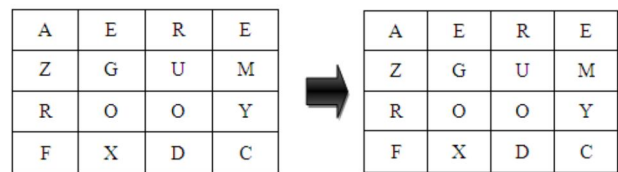
Board yang digunakan dalam *word game scramble* berukuran 4x4. Sedang kata-kata yang dicari dalam *board* 4x4 adalah kosakata dalam bahasa Inggris yang terdapat dalam kamus Inggris - Indonesia oleh Jhon M. Echols dan Hassan Shadily. Dan minimal kata yang didapat 2 karakter. Contoh : as, me, tv, at, dll.

KAJIAN PUSTAKA

Word Games Scramble

Scramble merupakan pengembangan dari permainan Scrabble yang sangat inspiratif, pola permainan ini yang menjadi inspirasi dalam mengerjakan tugas akhir ini. Permainan ini dimulai dengan mengacak huruf alphabet dengan kadar frekuensi setiap huruf berbeda untuk diisi

pada board berukuran 4x4, masing-masing grid diisi dengan huruf berbeda atau sama pada setiap isinya.



A	E	R	E
Z	G	U	M
R	O	O	Y
F	X	D	C

Gambar 1. Board 4x4 Word Game Scramble.

Randomization

Proses ini berfungsi sebagai pengacak huruf alfabet yang akan keluar di setiap node. Huruf-huruf pada setiap node diacak sedemikian rupa sehingga huruf-huruf umum muncul lebih sering dan lebih mudah untuk mendapatkan campuran yang baik antara vokal dan konsonan. Dari penelitian yang dilakukan oleh Andrea Krause berdasarkan Hidden Markov Models (HMM) menunjukkan bahwa huruf "E" adalah huruf yang paling sering keluar dalam 10.000 kata dari Reuters data set. Maka dalam tahap Randomization ini huruf "E" mendapat porsi lebih besar saat proses pengacakan. Berikut ini sekilas contoh pseudocode tentang alphabet generator untuk pengacakan huruf yang akan digunakan :

```
freq['a'] = 0.2 freq['b'] = 0.01 ...
freq['z'] = 0.02 Jumlah seluruh frekuensi diatas
adalah 1, maka dapat didefinisikan sebuah array dengan
interval seperti ini: intr['a'] = [0; 0.2)
intr['b'] = [0.2; 0.01) ... intr['z']
= [0.98; 1) Kemudian saat proses generalisasi
angka acak n pada interval [0; 1) program hanya mencari
dalam interval array dan mendapatkan huruf yang
terdapat pada interval tersebut. for(huruf = 'a'
.. 'z') if n in intr[huruf] then return
huruf;
```

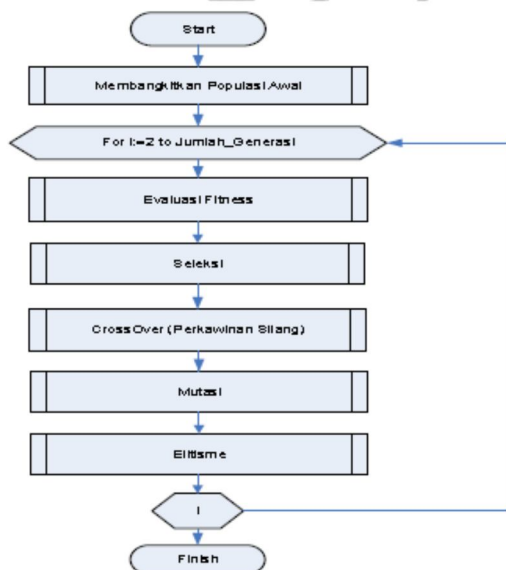
Algoritma Genetika

Algoritma Genetika adalah suatu algoritma pencarian berbasis pada mekanisme seleksi alam dan genetika. Pada dasarnya algoritma genetika mensimulasikan proses evolusi, dengan menghasilkan kromosom-kromosom dari tiap populasi secara random dan memungkinkan kromosom tersebut berkembang biak sesuai dengan hukum-hukum evolusi yang nantinya diharapkan akan dapat menghasilkan kromosom prima atau yang lebih baik. Kromosom ini merepresentasikan solusi dari permasalahan yang diangkat, sehingga apabila kromosom yang baik tersebut dihasilkan, maka diharapkan solusi yang baik dari permasalahan tersebut juga didapatkan.

Pendekatan yang diambil oleh algoritma ini adalah dengan menggabungkan secara acak berbagai pilihan solusi terbaik di dalam suatu kumpulan untuk mendapatkan generasi solusi terbaik berikutnya yaitu pada suatu kondisi yang memaksimalkan kecocokannya atau lazim disebut fitness. Generasi ini akan merepresentasikan perbaikan-perbaikan pada populasi awalnya. Dengan melakukan proses ini secara berulang, algoritma ini diharapkan dapat mensimulasikan proses

evolusioner. Pada akhirnya, akan didapatkan solusi-solusi yang paling tepat bagi permasalahan yang dihadapi. Proses yang ada pada algoritma genetika seperti gambar 2, meliputi:

- (1) Membuat atau membangkitkan populasi awal dari beberapa individu secara acak sehingga membentuk suatu populasi.
- (2) Menghitung nilai fitness dari generasi pertama
Nilai fitness adalah suatu nilai yang menyatakan baik atau tidaknya suatu solusi. Fitness yang semakin besar merupakan solusi yang paling baik karena nilai fitness diambil dari kebalikan nilai solusi ditambah bilangan yang mendekati nol (<0).
- (3) Melakukan proses seleksi individu menggunakan metode roulette wheel. Seleksi menggunakan metoda roulette wheel (roda roulette) akan memilih nilai dari fitness. Semakin tinggi nilai fitness maka semakin besar kemungkinan untuk terpilih.
- (4) Melakukan proses reproduksi dengan cara cross over
Cross Over merupakan salah satu komponen dalam algoritma genetika yang melibatkan dua induk untuk menghasilkan keturunan yang baru. Proses cross over dilakukan pada setiap individu dengan probabilitas tertentu biasanya mendekati 1, misalnya 0,8.
- (5) Melakukan proses mutasi dari individu terpilih
Mutasi merupakan operator yang menukar nilai gen dengan nilai kebalikannya dengan suatu probabilitas tertentu. Probabilitas mutasi diset sebagai $1/n$ dimana n adalah jumlah gen dalam individu atau kromosom. Gambar 6 merupakan flowchart dari proses mutasi
- (6) Elitisme yang berfungsi untuk menjaga agar individu bernilai fitness tertinggi tidak hilang selama proses evolusi dengan cara membuat copy-an dari individu dengan nilai fitness tertinggi tersebut.
- (7). Memasukkan individu ke populasi terbaru atau penggantian populasi. Semua individu dari suatu generasi digantikan sekaligus oleh sejumlah individu yang baru hasil pindah silang dan mutasi.



Gambar 2. Flowchart Algoritma Genetika

METODE REKAYASA

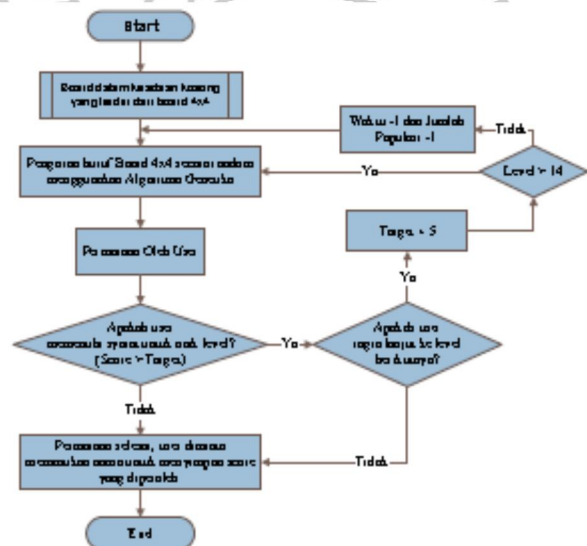
Analisa Sistem

Aplikasi *scramble* adalah permainan untuk mencari kosakata sebanyak mungkin yang terdapat dalam *board* 4x4. Peraturan yang terdapat dalam permainan *scramble*, kosakata yang dicari minimal 2 karakter, pencarian kosakata tidak boleh melompati grid *board* 4x4, tidak boleh mengulang kosakata yang sudah ditemukan walaupun dengan tempat yang berbeda.

Aturan penilaian, setiap huruf yang berhasil didapat bernilai 5 poin dan di level 1 harus mendapatkan minimal 10 huruf (50 poin) dalam waktu 120 detik untuk bisa melanjutkan ke level selanjutnya. Setiap naik level target bertambah 5 poin dan waktu berkurang 5 detik (untuk level 15 ke atas waktu tetap tidak akan dikurangi). Untuk batas level tidak terbatas karena semakin tinggi level yang dicapai semakin susah kata susunan kosakata dalam *board* 4x4.

Jumlah populasi pada proses pengacakan algoritma genetika dikurangi satu (jumlah populasi - 1) untuk setiap naik level, agar tingkat optimalitas hasil pengacakan berkurang sehingga kosakata lebih susah untuk ditemukan dan pengurangan jumlah populasi ini maksimal level 15 (level 1 menggunakan 20 jumlah populasi).

Dalam proses algoritma genetika ada istilah yang disebut gen dimana gen tersebut adalah nilai yang menyatakan satuan dasar (dalam proses ini gen tersebut berupa huruf alphabet), individu adalah gabungan dari gen yang berupa nilai (contoh : PEACEASCDTERBNI), populasi adalah sekumpulan individu dan generasi adalah sekumpulan individu terbaik yang diperoleh dari populasi.



Gambar 3. Flowchart Word Game Scramble

Gambar 3 adalah *flowchart* jalannya sistem *word game scramble* yang akan dilakukan. Mulai dari proses pengacakan dengan algoritma genetika hingga mencetak sebuah hasil akhir dari *score* yang diperoleh pemain (*user*) sesuai dengan aturan-aturan yang telah diberikan. Desain

Desain Board

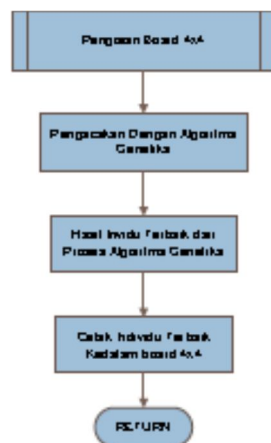
Dalam desain *board* yang akan dibuat dalam sistem *word game scramble* adalah *board* 4x4, seperti gambar 4. Awal permainan *board* 4x4 ini dalam keadaan kosong karena huruf akan terisi secara acak ke dalam *board* setelah pemain (*user*) memulai permainan *scramble*.



Gambar 4. Desain Board 4x4

Proses Pengisian Board dengan Algoritma Genetika

Pada proses ini *board* akan di isi dengan huruf alphabet yang dilakukan secara acak dan proses pengacakan dilakukan menggunakan algoritma genetika. Dari hasil proses algoritma genetika akan didapatkan individu terbaik. Selanjutnya individu terbaik tersebut akan dicetak pada board 4x4. Langkah-langkah proses ini seperti pada gambar 5.



Gambar 5. Flowchart Pengisian Board 4x4

Tahap-tahap proses pengacakan *word game scramble* dengan algoritma genetika. nya adalah sebagai berikut:

1) Membangkitkan Populasi Awal

Proses pembangkitan populasi awal ini dilakukan dengan cara mengacak gen-gen yang secara *random* (gen yang diambil terdiri dari huruf alfabet). Setelah itu gen di ambil sebanyak 16 gen karena *board* 4x4 kemudian akan digabung menjadi sebuah individu.

2) Evaluasi Fitness

Nilai Fitness merupakan suatu nilai yang menyatakan baik atau tidaknya suatu individu. Nilai Fitness ini akan dijadikan acuan dalam mencapai nilai optimal dari aturan-aturan yang telah ditentukan. Aturan yang dibuat pada dalam proses evaluasi *fitness* ini menggunakan teori Andrea Krause berdasarkan *Hidden Markov Models* (HMM) seperti pada tabel 5 dimana setiap huruf mempunyai bobot tersendiri. *Fitness* yang semakin besar merupakan solusi yang paling baik. Nilai *fitness* yang terbentuk

pada satu individu akan dibandingkan dengan nilai *fitness* dari individu lain sehingga akan didapatkan individu yang terbaik.

Tabel 1. Bobot Huruf Menurut HMM

No. Urut	Huruf	Frekuensi	Inisialisasi Bobot
1	E	11.1607 %	11
2	A	8.4966 %	8
3	R	7.5809 %	7
4	I	7.5448 %	7
5	O	7.1635 %	7
6	T	6.9509 %	7
7	N	6.6544 %	7
8	S	5.7351 %	6
9	L	5.4893 %	5
10	C	4.5388 %	5
11	U	3.6308 %	4
12	D	3.3844 %	3
13	P	3.1671 %	3
14	M	3.0129 %	3
15	H	3.0034 %	3
16	G	2.4705 %	2
17	B	2.0720 %	2
18	F	1.8121 %	2
19	Y	1.7779 %	2
20	W	1.2899 %	1
21	K	1.1016 %	1
22	V	1.0074 %	1
23	X	0.2902 %	1
24	Z	0.2722 %	1
25	J	0.1965 %	1
26	Q	0.1962 %	1

3) Seleksi

Pada tahap seleksi ini ialah mencari nilai *fitness* dari semua individu pada suatu populasi yang mempunyai nilai *fitness* terbesar maka akan terpilih, dan apabila nilai *fitness* terbesar ada yang sama maka akan diambil satu nilai *fitness* saja secara *random*. Gambar 8 merupakan Flowchart Seleksi.

4) Cross Over

Cross over dilakukan dengan single string, dimana langkah pertama yang di ambil adalah mencari dua node secara *random* sebagai penentu titik dimana akan menjadi proses *cross over*. Dua node yang diambil secara acak tidak boleh sama kedua node tersebut akan diacak kembali sehingga tidak menghasilkan node yang sama dan index node pertama harus lebih kecil dari index node kedua apabila index node pertama lebih besar dari index node kedua akan diubah secara otomatis. *Cross over* dilakukan dengan cara menukarkan seluruh gen yang terpilih mulai dari gen awal hingga gen terakhir.

5) Mutasi

Mutasi juga dilakukan dengan single string sama seperti *Cross Over*, yang membedakan antara *cross over* dan *mutasi* adalah proses pertukaran posisi dari antar node yang terpilih secara *random*. Pada *mutasi* menukarkan posisi tempat gen yang terpilih saja tanpa menukar

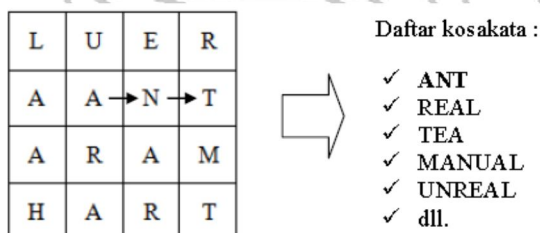
deret antar node. Sehingga besar kecilnya node *index* tidak berpengaruh dalam proses *mutasi*.

6) *Elitisme*

Selanjutnya mencari nilai *fitness* dari individu terbaik. Dari nilai *fitness* terbaik maka akan dilakukan proses *elitisme*. Tujuan dari *elitisme* agar individu terbaik yang telah terpilih tersebut tidak hilang dalam proses regenerasi. Karena arti dari *elitisme* adalah menyimpan individu terbaik setiap regenerasi. Dari hasil proses *seleksi* dari *elitisme* deret individu akan di *generate* kedalam *board* 4x4,

Permainan

Dalam tahap ini pemain (*user*) mencari kosakata dalam *board* 4x4 yang sudah dicetak. Dalam tahap pencarian kosakata pemain (*user*) harus sesuai dengan peraturan yang ditentukan. Kosakata yang berhasil ditemukan oleh pemain (*user*) akan di cek terlebih dahulu apakah kosakata yang ditemukan ada dalam database. Apabila kosakata yang ditemukan ada dalam database, *word game scramble* akan menilai jumlah kata yang berhasil ditemukan sesuai dengan panjang kata yang diperoleh. Semakin panjang dan banyak kosakata yang diperoleh maka akan semakin tinggi *score* yang didapat. Gambar 6 adalah gambar pencarian kosakata pada *board* 4x4.



Gambar 6. Pencarian Kosakata Pada Board 4x4

Daftar Nilai Tertinggi.

Proses ini adalah proses terakhir saat permainan *word game scramble* selesai dimana *score* yang diperoleh oleh pemain (*user*) akan disimpan dan sebelum disimpan pemain (*user*) diminta untuk memasukkan nama. Daftar nilai akan di urutkan dari *score* yang tertinggi dimana daftar *score* yang ditampilkan hanya 10 daftar saja, sehingga apabila *score* yang diperoleh masih dibawah 10 daftar *score* yang sudah ada, nilai tidak akan ditampilkan. Gambar 19 adalah gambar *flowchart high score*.

PEMBAHASAN

Aplikasi *word game scramble* terdiri dari *board* 4x4 dimana untuk melakukan permainan ini harus mencari kosakata yang terdapat pada *board* 4x4 yang susunan katanya diacak dengan menggunakan algoritma genetika. Untuk dapat melanjutkan permainan ke level berikutnya harus memenuhi target yang sudah ditentukan dalam waktu yang disediakan agar memperoleh *score* tertinggi dan setiap level waktu yang disediakan semakin singkat.

Pada permainan ini, pemain harus mencari kata yang akan di acak pada proses *randomization* menggunakan algoritma genetika kemudian komputer akan mencocokkan pada kamus apakah kata yang ditemukan oleh pemain *valid* atau terdapat pada database tersebut. Aturan pada pencarian kata oleh pemain sebagai berikut (1) Sebuah kata yang didapat terdiri dari 2 huruf atau lebih (2) Kata yang sudah ditemukan tidak dapat diulangi (walaupun terdapat kata yang sama namun dengan path yang berbeda). (3) Jalan yang ditelusuri oleh huruf-huruf pada kata yang dipilih harus terhubung secara horisontal, vertikal, atau diagonal. Pemain tidak dapat melompati grid untuk mendapatkan huruf berikutnya karena dalam *board* grid yang akan dilompati otomatis akan *disable*. (4) Dalam level 1 pemain (*user*) harus mendapatkan minimal 10 huruf (50 poin) dalam waktu 120 detik untuk bisa melanjutkan ke level berikutnya, dan setiap naik level target bertambah 5 poin dan waktu berkurang 5 detik.

Kata yang ditemukan oleh pemain mempunyai nilai yang berbeda sesuai dengan jumlah huruf pada kata yang sudah masuk dalam daftar kata yang ditemukan pemain. Dengan nilai – nilainya sebagai berikut:

Tabel 2 Tabel Scoring

Panjang Kata	Nilai
2	10
3	15
4	20
5	25
6	30
7	35

Pada gambar 7 adalah tampilan Menu Utama setelah pemain (*user*) menekan tombol *start* (F2), *frame timer* yang berisi Sisa Waktu yang berisi angka 0 detik secara otomatis akan berubah menjadi 120 detik dan dihitung mundur.



Gambar 7. Tampilan Menu Utama Setelah di Eksekusi

Dengan sisa waktu yang terus berjalan pemain (*user*) harus mencari kosakata minimal 10 huruf dikarenakan target setiap level harus mencapai score 50 sehingga user dapat melanjutkan kelevel berikutnya.

Pada saat user melakukan pencarian kosakata dan kosakata yang dicari ada pada database tampilan menu utama setelah di tekan tombol generate akan muncul *MessageBox* seperti pada gambar 8, kemudian secara

otomatis *TextBox* kosakata akan terisi kosakata yang ditemukan oleh pemain (*user*) dan *TextBox* tejemahan akan berisi arti dari kosakata tersebut, seperti pada gambar 9 sehingga pemain (*user*) dapat memperdalam pengetahuan kosakata dalam bahasa Inggris.



Gambar 8. Tampilan Menu Utama Saat Kosakata yang Dicari Benar.



Gambar 9. Menu Utama Menampilkan Kosakata dan Terjemahannya.

Pemain (*user*) tidak boleh mengulang kosakata yang sudah ditemukan meskipun dalam posisi susunan kata yang berbeda,

Apabila pemain (*user*) sebelum waktu habis menemukan kata minimal 10 total huruf maka pemain (*user*) berhasil menuju ke-level berikutnya, Dan begitu juga sebaliknya, apabila pemain (*user*) sampai waktu habis tidak dapat menemukan kosakata sesuai yang ditargetkan maka pemain (*user*) tidak dapat menuju ke-level berikutnya dan game otomatis akan berakhir.

Kemudian pemain (*user*) akan diminta memasukkan nama untuk disimpan pada *form high score*. Setelah pemain (*user*) selesai memasukkan nama kemudian *form high score* otomatis akan keluar dan menampilkan daftar nama pemain (*user*) yang berhasil memperoleh nilai terbaik, dan *score* tertinggi akan mendapatkan posisi pertama. Untuk lebih jelasnya bisa di lihat pada gambar 38.

No	Nama	Score
1	ANDREYAN	1630
2	VERRY	1080
3	NARYTA	800
4	RAMA	640
5	SANTI	440
6	TRIAL	160
7	RAMA	85
8	UCUP	80
9	CECE	45
10	ZAKARIA	30

Reset High Score

Gambar 10. Tampilan *High Score*

Hasil Analisa Proses Algoritma Genetika

- (1) Proses Algoritma Genetika dengan 5 Generasi dan 5 Populasi.

Hasil yang didapat dalam proses pengacakan ini adalah deret individu yang dicetak dalam *board 4x4* kurang optimal karena jumlah nilai *fitness* dari deret individu tidak mencapai angka 100, sehingga kosakata yang lebih susah ditemukan.

Proses GA

Populasi Awal

BJEUIAFWOKVAQMER = 75	Jmlh Generasi	5
RQIHHEXWEYBINEEF = 87	Jmlh Populasi	5
JSTOBAFIILAKQXIE = 81		
SKOTPIFAABWAAMAQ = 80		
TMECLBXDEIHIRFAP = 85		
TSIBOWHKEVCYGPPEF = 71		

E = 14, X = 6, Node Cross

Cross Over: RQIHHEENIBYEWKF

Seleksi: RQIHHEXWEYBINEEF = 87

E = 6, X = 14, Node Mutasi

Elitisme Setiap Generasi

FTALIKFAXAQAOREI = 90
CRGIOYAFMMNTALAA = 89
DOEQIOSWATALOMJN = 89
MAAGTLRSVMNAYNEB = 87
JFIPMAMSAYEECOHR = 87
RQIHHEXENIBYEWEF = 87

Seleksi dari Elitisme: FTALIKFAXAQAOREI = 90

Mutasi: RQIHHEXENIBYEWEF

Random GA: Start

Random Biasa: Start

Gambar 11. Proses dengan 5 Generasi dan 5 Populasi

- (2) Proses Algoritma Genetika dengan 5 Generasi dan 10 Populasi.

Hasil yang didapat bisa dikatakan optimal karena jumlah nilai *fitness* dari deret individu sudah mencapai angka 102, sehingga kosakata yang lebih mudah ditemukan.

Proses GA

Populasi Awal

MYISMIDFARGIJXOU = 70
OKEGAIMRIAUBHSAJ = 85
NCEWVIPZITLITPAD = 83
RGIZLINDEYTSVNAT = 88
ZMINJAHQIBMIYRAK = 68
FZAOCEJGEMHINDAB = 81
BNAKGARPIIZIMCEV = 80
JOACNAFREWOEPFIL = 93
REEZVIYXWHLEZMMK = 69

Jmlh Generasi 5
Jmlh Populasi 10

T = 6
E = 5
Node Cross

Cross Over
NUIUNTEBELHITFEJ

Seleksi
NUIUNTEBELHITFEJ = 96

L = 9
E = 14
Node Mutasi

Elitisme Setiap Generasi

ZOEJETCPTARAWWS = 95
MAAEEMQEIEXYVSCG = 91
SNEHDEOJAKGTWET = 93
KDAREEBEDAWRZETN = 99
GTAECEKEADNEMLAQ = 102
NUIUNTEBEEHITFLJ = 96

Seleksi dari Elitisme
GTAECEKEADNEMLAQ = 102

Mutasi

NUIUNTEBEEHITFLJ

Random GA
Start

Random Biasa
Start

Gambar 11. Proses dengan 5 Generasi dan 10 Populasi

- (3) Proses Algoritma Genetika dengan 10 Generasi dan 5 Populasi.

Hasil yang didapat bisa dikatakan optimal karena jumlah nilai *fitness* dari deret individu sudah mencapai angka 101 dan tidak jauh berbeda dengan hasil 5 generasi, 10 populasi sehingga kosakata yang lebih mudah ditemukan.

Proses GA

Populasi Awal

HMIGVCWQJIZPEZ = 59
HSAJWMFVIZJELZJG = 54
YJEHBKEDAQNWTAER = 84
KGETOINOEVEZFLXU = 85
ZQWKRASPEPLICFAD = 72
OLIBFAZGAXRIFIO = 80

Jmlh Generasi 10
Jmlh Populasi 5

G = 1
Z = 10
Node Cross

Cross Over
KZVEONIOEGEFLXU

Seleksi
KGETOINOEVEZFLXU = 85

E = 11
V = 2
Node Mutasi

Elitisme Setiap Generasi

JRAAXEKSECIERZA = 101
AKICUATFIUBBXIS = 82
CBPSPILDYDEELLK = 79
SQAQHAZBETNANGIR = 86
SCEFZEMWEEGIFDIO = 90
OICMAIZNEAHEHXIE = 100

Seleksi dari Elitisme
JRAAXEKSECIERZA = 101

Mutasi

KZVEONIOEGVFLXU

Random GA
Start

Random Biasa
Start

Gambar 12. Proses dengan 10 Generasi dan 5 Populasi

- (4) Proses Algoritma Genetika dengan 10 Generasi dan 10 Populasi.

Hasil yang didapat dalam proses pengacakan ini adalah deret individu yang dicetak dalam *board* 4x4 optimal karena jumlah nilai *fitness* dari deret individu sudah mencapai angka 107, sehingga kosakata yang lebih mudah ditemukan.

Proses GA

Populasi Awal

HNIZZKFVYMCRAHMD = 58
GRIPILMVRKQALXAW = 67
FWEPVEOYESZECQEB = 86
ASANBHFLWFENMEE = 88
SJIULEGHFTBEMKIQ = 73
IRWSVAADIYIPNYS = 82
NKIWEHLEFHIPFAM = 75
BCIFLIBYKOMEFPCZ = 65
NTIIVDXLAIPETEEG = 98

Jmlh Generasi 10
Jmlh Populasi 10

E = 11
T = 12
Node Cross

Cross Over
NTIIVDXLAIPETEEG

Seleksi
NTIIVDXLAIPETEEG = 98

E = 13
T = 11
Node Mutasi

Elitisme Setiap Generasi

FJEMOPECSTRASKIA = 89
FELTEACEULZQNER = 107
NYEAEABJEYLAGAE = 105
DNPAREAUSQLIEQEP = 96
MCEPEWEAIHEXFOER = 102
DCESNANEOEHAGJRI = 104

Seleksi dari Elitisme
FELTEACEULZQNER = 107

Mutasi

NTIIVDXLAIPETEEG

Random GA
Start

Random Biasa
Start

Gambar 13. Proses dengan 10 Generasi dan 10 Populasi

Hasil Analisa Kosakata yang Diperoleh

- (1) Pengacakan dengan Algoritma Genetika

Hasil yang diperoleh dari pengacakan menggunakan algoritma genetika diperoleh kosakata dalam *board* 4x4 akan lebih banyak, sehingga *user* lebih mudah untuk mencari kosakata dalam bermain *word game scramble*. Gambar 14 adalah proses penemuan kosakata dalam *board* 4x4 hasil pengacakan dengan algoritma genetika mengambil jumlah populasi 17 dan jumlah generasi 5.

Hasil Kosakata

Di Temukan 86 Kata

acre
acreage
aec
age
aged
agree
arc
are
area
arcs

Gambar 14. Penemuan Kosakata dengan Algoritma Genetika

- (2) Pengacakan Biasa

Hasil yang diperoleh dari pengacakan menggunakan pengacakan biasa diperoleh kosakata dalam *board* 4x4 lebih akan lebih sedikit bahkan dapat terjadi dalam suatu pengacakan tidak terdapat kosakata satu pun yang di dapat. Gambar 46 adalah proses penemuan kosakata dalam *board* 4x4 hasil pengacakan biasa.



Gambar 16. Penemuan Kosakata dengan Pengacakan Biasa

Tim Penyusun, 2005. Pedoman Tugas Akhir. Surabaya: Unesa University Press.

Winoto Dwi Suryo. 2009. Aplikasi penjadwalan mata kuliah menggunakan algoritma genetika (studi kasus jurusan teknik elektro unesa). Tugas Akhir tidak dipublikasikan, Surabaya : Universitas Negeri Surabaya.

PENUTUP

Simpulan

Dari hasil uji coba yang telah dilakukan pada aplikasi *word game scramble* dengan algoritma genetika, maka dapat diambil kesimpulan :

1. Dari hasil pengacakan yang dilakukan dengan algoritma genetika untuk jumlah populasi dan jumlah generasi berbanding lurus karena semakin besar jumlah populasi dan jumlah generasi hasil individu yang didapat semakin baik.
2. Hasil pengacakan dari proses algoritma genetika dengan pengambilan data huruf hasil penelitian Andrea Krausa berdasarkan *Hidden Markov Models* didapatkan lebih banyak kosakata dalam *board* 4x4 dibandingkan dengan hasil pengacakan biasa dimana dalam hasil pengacakan biasa kosakata yang dihasilkan sangat tidak optimum dan sering kali pada tahap pengacakan tersebut tidak terdapat kosakata yang dihasilkan sama sekali.

DAFTAR PUSTAKA

El-Bahri, M. Agus. S. Desember 2008. Teaching English Vocabulary Using Games. <URL: <http://www.scribd.com/doc/9639825/>> (diakses tanggal 04 Juni 2011)

Krause, Andreas, dan Zollmann, Andreas. 2002. Not So Randomly Typing Monkeys – Rank-frequency Behavior of Natural and Artificial Languages, Algorithms for Information Networks – Project Report.

Sabdifha Noswa. 2010. Perancangan dan Pembuatan Aplikasi Word Games Scramble pada Board Dua Dimensi. Tugas Akhir tidak dipublikasikan, Surabaya : Institut Teknologi Sepuluh Nopember.

Sri Kusumadewi. 2003. Artificial Intelligence (Teknik dan Aplikasinya). Yogyakarta: Graha Ilmu.

Suyanto. 2005. Algoritma Genetika dalam Matlab. Yogyakarta: Andi.